Two-machine Jobshop with Controllable Processing Times

Kailiang Xu*[1], Sha Liu[2]

School of Automation and Information Engineering, Xi'an University of Technology

*[1]klxu@xaut.edu.cn; [2]shaliu.gyx@gmail.com

Abstract

This paper concerns on a two-machine job-shop scheduling problem, where the processing time of the operations is controllable, and is modeled by a non-linear convex resource consumption function. The objective is to minimize the resource consumption under the condition that makespan should not exceed the given deadline. The problem is strongly NP-hard. In order to solve this problem, a polynomial time resource allocation algorithm is presented, which optimally allocate resource to job operations with arbitrarily given processing order. After this, a heuristic algorithm is presented, which provides optimal or near-optimal processing sequence for the job operations.

Keywords

Two-machine Jobshop; Convex Resource Consumption Function; Heuristic

Introduction

The job-shop problem is well-studied in literature. In most research work, it is assumed that processing time of job operations is constant and known in advance. However, in many practical environments, processing time can be controlled by the consumption of an extra resource, such as gas, fuel, electricity power, cash and labors. For example, Janiak (Janiak, 1989) described an application in steel mills, where batches of ingots have to be preheated before being hot-rolled in a blooming mill, and both the preheating time and the rolling time are inversely proportional to the gas flow intensity. Therefore, the classical job-shop scheduling problem needs to be generalized with controllable processing time of the job operations.

In this paper we study a two-machine job-shop scheduling problem where processing time and resource consumption are modeled by a non-linear convex function introduced by Monma et al. (Monma et al., 1990) and Shabtay and Steiner (Shabtay & Steiner, 2007), stated formally as follows:

$$p_{j,i}(u_{j,i}) = \left(\frac{\omega_{j,i}}{u_{j,i}}\right)^k \qquad j=1,2,\ldots,n \quad i=1,2,\ldots,m \quad (1)$$

where $\omega_{j,i} > 0$ represents the workload of operation $O_{j,i}$ that belongs to job j and is processed on machine i, $u_{j,i} > 0$ is the amount of the resource allocated to the operation, and k is a constant positive parameter.The problem is formally stated as follows: A set of jobs $\mathrm{J} = \{1,2,\ldots,n\}$ are to be processed in a two-machine job-shop without recirculation. Each job has at most two operations, $O_{j,1}$ and $O_{j,2}$. The processing time of each operation, $p_{j,1}$ or $p_{j,2}$, is a decreasing non-linear convex function with respect to the resource consumption $u_{j,1}$ or $u_{j,2}$ described by Eq.1. The objective is to determine the processing sequence as well as the resource allocation of the operations on both machines, such that the makespan is limited within deadline K, while the total resource consumption is minimized. Using the three field problem classification introduced by Graham et al. (Graham et al. 1979)and extended by Shabtay and Steiner(Shabtay & Steiner, 2007), this problem can be denoted as $J_2 \mid conv, C_{\max} \le K \mid \sum_{m=1}^{2}\sum_{j=1}^{n} u_{j,m}$.

Optimal Resource Allocation

According to the equivalent workload method defined by Monma et al., when n operations are to be processed in serial with makespan no larger than deadline K, the optimal solution is

$$\begin{aligned} p_j &= \frac{\omega_j^{\frac{k}{k+1}}}{\sum_{i=1}^{n}\omega_i^{\frac{k}{k+1}}}K \qquad j=1,2,\ldots,n \\ u_j &= \omega_j^{\frac{k}{k+1}}\left(\sum_{i=1}^{n}\omega_i^{\frac{k}{k+1}}\right)^{\frac{1}{k}}K^{-\frac{1}{k}} \qquad j=1,2,\ldots,n \qquad (2) \\ U &= \left(\sum_{j=1}^{n}\omega_j^{\frac{k}{k+1}}\right)^{\frac{k+1}{k}}K^{-\frac{1}{k}} \end{aligned}$$

and operations can be regarded as a single one with

equivalent workload $\Omega_s = \left(\sum_{i=1}^{n} \omega_i^{\frac{k}{k+1}} \right)^{\frac{k+1}{k}}$. When n operations are to be processed in parallel, the optimal solution is

$$\begin{aligned} p_j &= K \qquad j = 1,2,\ldots,n \\ u_j &= \omega_j K^{-\frac{1}{k}} \qquad j = 1,2,\ldots,n \\ U &= \sum_{i=1}^{n} \omega_i K^{-\frac{1}{k}} \end{aligned} \tag{3}$$

and operations can also be regarded as a single operation with equivalent workload $\Omega_p = \sum_{i=1}^{n} \omega_i$.

A job has at most two operations, and they must be processed in sequence. Suppose resource has been optimally allocated to the operations. If the second operation of a job starts immediately after the first completes, the precedence constraint between them is named as a *critical precedence constraint*. Otherwise, the constraint is named as *slack precedence constraint*, because the relaxation of the constraint has no influence on the resource allocation.

In the schedule, operations are divided into sections by critical precedence constraints. Within each section, there exist only slack constraints. Since slack constraints have no influence on the resource allocation, the equivalent workload method can be applied. For example, suppose operations are divided into sections $\Lambda = \{\Lambda_1, \Lambda_2, \ldots, \Lambda_m\}$, where Λ_l is bounded by the precedence constraints of job i and job j. According to the equivalent workload method, operations within section Λ_l can be regarded as a single operation with equivalent workload

$$\Omega_{\Lambda_l} = \Omega_{\Lambda_l,1} + \Omega_{\Lambda_l,2} = \left(\sum_{O_{h,1} \in \Lambda_l} \omega_{h,1}^{\frac{k}{k+1}} \right)^{\frac{k+1}{k}} + \left(\sum_{O_{h,2} \in \Lambda_l} \omega_{h,2}^{\frac{k}{k+1}} \right)^{\frac{k+1}{k}} \tag{4}$$

while the equivalent workload of all the operations is

$$\Omega_\Lambda = \left(\Omega_{\Lambda_1}^{\frac{k}{k+1}} + \ldots + \Omega_{\Lambda_m}^{\frac{k}{k+1}} \right)^{\frac{k+1}{k}} \tag{5}$$

Given deadline K, the processing time of section Λ_l can be calculated as

$$p_{\Lambda_l} = \left(\frac{\Omega_{\Lambda_h}}{\Omega_\sigma} \right)^{\frac{k}{k+1}} K \tag{6}$$

and the processing time of the operations within the section is

$$p_{1,l} = \left(\frac{\omega_{1,l}\, \Omega_{\Lambda_h}}{\Omega_{1,\Lambda_h}\, \Omega_\sigma} \right)^{\frac{k}{k+1}} K \qquad i+1 \le l \le j \tag{7}$$

$$p_{2,l} = \left(\frac{\omega_{2,l}\, \Omega_{\Lambda_h}}{\Omega_{2,\Lambda_h}\, \Omega_\sigma} \right)^{\frac{k}{k+1}} K \qquad i \le l \le j-1 \tag{8}$$

Therefore, the problem left is to determine the critical precedence constraints that optimally divide operations into sections.

In order to solve this problem, operations are transformed into an acyclic directed graph, as Fig.1 shows, where each node represents a precedence constraint between two operations. Draw arcs between nodes. Each arc represents a section bounded by the precedence constraints represented by the emanating and sinking node of the arc. Assume both constraints are critical, and there exists no other critical constraint between them. Calculate the equivalent workload as well as the relative processing times of the operations within section $A_{i,j}$ according to Eq.4 and Eq.7 and Eq.8.

Because the calculation assumes the relaxation of the precedence constraints, it is necessary to check the feasibility of the result. For every operation that has a predecessor also belonging to $A_{i,j}$, check their start time and completion time. If none of these operations starts earlier before its predecessor completes, the calculation is feasible, and let the equivalent workload be the length of the arc. Otherwise, the calculation is infeasible, and the arc is removed from the graph.

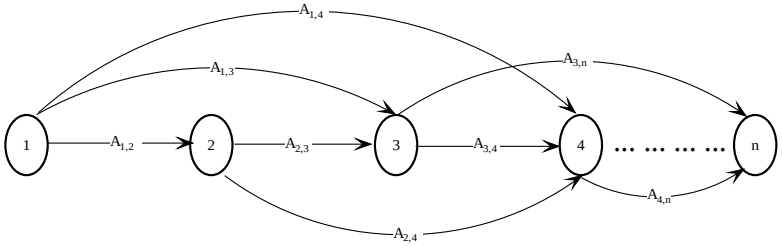

FIG. 1 THE DIRECTED GRAPH THAT CONTAINS ALL THE FEASIBLE SECTIONS OF THE OPERATIONS

After all the feasible arcs are enumerated, there exist a variety of paths from node U to node S. Each path represents a feasible solution to the resource allocation problem, and the equivalent workload of the solution is

$$\Omega = \left(\sum_{A_{i,j} \in P} \Omega_{i,j}^{\frac{k}{k+1}} \right)^{\frac{k+1}{k}} \quad (9)$$

where P is a path from node U to node S. Obviously, there exists at least one path that has the minimum equivalent workload, which can be found using a dynamic programming method.

Based on the above discussion, the optimal resource allocation algorithm is presented in the follows:

Algorithm 1. Optimal resource allocation

1) Enumerate precedence constraints in the schedule. Suppose there are m precedence constraints, let nodes $1,2,\ldots,m$ represent these constraints. Let node 0 and $m+1$ represent the start node and end node;

2) For each $j = 1,2,\ldots,m+1$, draw arcs $A_{i,j}$ for $i = 0,1,\ldots,j-1$. Calculate the relative processing time of the operations in the arcs according to Eq.7 and Eq.8. Check the feasibility of each arc by examining if no operation starts earlier than its predecessor. If the arc is feasible, calculate the equivalent workload according to Eq.4, and let the value be the length of the arc. If the arc is infeasible, remove the arc from the graph;

3) For each $j = 1,2,\ldots,n$, calculate Ω_j using the above recursive equations.

4) Search for the shortest path from node 0 to node n, where each arc on the path represents a section of the operations in the optimal solution. Calculate processing time of the operations using Eq.7 and Eq.8.

Properties of the Optimal Solution

Some of the properties of the problem will be discussed in this section. For convenience, operations are classified into three subsets:

1) O_1: Operations that have a successor;

2) O_2: Operations that have a predecessor;

3) O_3: Operations that have no successor or predecessor.

Property 1. There exists an optimal schedule, where all the operations in O_1 are scheduled first on machine 1 and machine 2.

Property 2. There exists an optimal schedule, where all the operations in O_3 are scheduled before

Proof: The property can be proved by assuming every operation in O_3 to be the predecessor of a dummy operation with processing time equal to 0.

Property 3. There exists an optimal schedule, where operations in O_2 are scheduled in the same order as their predecessors.

Property 4. In any a solution generated according to property 1-3, if not all the precedence constraints are slack, then jobs either in $J_{1,2}$ or $J_{2,1}$ have no critical precedence constraint (unless the job is the only member of the set).

The Heuristic Method

For our problem, although the processing time is unknown, it has a close relation to the workload of the operations. Therefore, a heuristic method named SWL(1)-LWL(2) is proposed in this section, which can be stated as follows:

Algorithm 2. SWL(1)-LWL(2) method

1) Let $J_{1,2}$ denote the set of the jobs that are processed on machine 1 first, while $J_{2,1}$ the set of the jobs processed on machine 2 first. Jobs with only one operation are not considered;

2) Sort jobs in $J_{1,2}$ as follows: Jobs whose first operation has smaller workload than the second operation are ordered first, while others are ordered second. For the first group, a job whose first operation has smaller workload is ordered first. For the second group, a job whose second operation has larger workload is ordered first;

3) Sort operations in $J_{2,1}$ in the same manner;

4) Schedule the first operations of the jobs in $J_{1,2}$ on machine 1 in their sorted order. Schedule the first operations of the jobs in $J_{2,1}$ on machine 2 in their sorted order;

5) Schedule jobs with only one operation on machine 1 and machine 2 in arbitrary order;

6) Schedule all the left operations on machine 1 and machine 2 in the same order as their predecessors;

7) Calculate resource allocation of the operations.

Although the problem is in theory strongly NP-hard, the SWL(1)-LWL(2) method is very likely to generate the optimal solution. If this is not the case, the solution is normally close to the optimum, and a branch and bound or a tabu-search algorithm can further improves it.

Conclusion

A two-machine jobshop scheduling problem with controllable processing times is studied in this paper. A polynomial time resource allocation algorithm is presented first, which minimizes the resource consumption under the condition that the processing sequence of the operations is determined. After that, several properties of the optimal schedule are discussed, and a heuristic algorithm for optimal or near-optimal processing sequence is presented.

ACKNOWLEDGMENT

This research is supported by National Natural Science Foundation of China (No.61203183).

REFERENCES

Grabowsk A., Janiak A. "Job-shop scheduling with resource-time models of operations." European Journal of Operations Research 28 (1987): 58-73.

Graham RL, Lawler EL, Lenstra JK, Rinnooy Kan AHG. "Optimization and approximation in deterministic sequencing and scheduling: a survey." Annals of Discrete Mathematics 5(1979): 287-326.

Janiak A. "Minimization of the blooming mill standstills-mathematical model, suboptimal algorithms." Mechanika 8 (1989): 37-49.

Janiak A, Szkodny T. "Job-shop scheduling with convex models of operations." Methematical and Computer Modelling 20 (1994) : 59-68.

Monma CL., Schrijver A., Todd MJ., Wei VK. "Convex resource allocation problems on directedacyclic graphs: duality, complexity, special cases and extensions." Mathematics of Operations Research 15 (1990): 736-748.

Shabtay D, Kaspi M, Steiner G. "The no-wait two-machine flow-shop scheduling problem with convex resource-dependent processing times." IIE Transaction 39(2007): 539-557.

Shabtay D, Steiner G. "A survey of scheduling with controllable processing times." Discrete Applied Mathematics 155(2007): 1643-1666.

Xu K, Feng Z, Jun K. "A tabu-search algorithm for scheduling jobs with controllable processing times on a single machine to meet due dates." Computers and Operations Research 37(2010) : 1924-1938.